EXPERIMENT - 22

Object Oriented Programming Lab

Aim

Implement a class string containing the following functions:

a) Overload + operator to carry out the concatenation of strings.
b) Overload = operator to carry out string copy.
c) Overload <= operator to carry out the comparison of strings.
d) Function to display the length of string.
e) Function tolower() to convert upper case to lower case.
f) Function toupper() to convert lower case letters to upper case.

Syeda Reeha Quasar
14114802719
4C7

EXPERIMENT – 22

Aim:

Implement a class string containing the following functions:

a. Overload + operator to carry out the concatenation of strings.
b. Overload = operator to carry out string copy.
c. Overload <= operator to carry out the comparison of strings.
d. Function to display the length of string.
e. Function tolower() to convert upper case to lower case.
f. Function toupper() to convert lower case letters to upper case.

Overload + operator to carry out the concatenation of strings.

Source Code:

```
#include <iostream>
#include <cstring>

using namespace std;

// concatenating 2 strings
class concatString{
    public:
        char str[100]; // class object

        concatString(){} // no parameter constructor

        // initalising class variable
        concatString(char s[]){
            strcpy(this->str, s);
        }

        // overloading operator+ for concatenation
        concatString operator+(concatString& s2){
            concatString s3;
            strcat(this->str, s2.str);
            strcpy(s3.str, this->str);
            return s3;
        }
```

```
};

int main(){
    char s1[100], s2[100];

    cout << "Enter 2 strings u want to concatenate" << endl;
    cin >> s1 >> s2;

    concatString a1(s1);
    concatString a2(s2);
    concatString a3;

    a3 = a1 + a2;

    cout << "concatenation: " << a3.str;

    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
reeha
saba
concatenation: reehasaba
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
the
world
concatenation: theworld
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
Hello
World!
concatenation: HelloWorld!
PS D:\sem 4\cpp\oops>
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
reeha
saba
concatenation: reehasaba
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
the
world
concatenation: theworld
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ concatStr+.cpp -o concatStr+ } ; if ($?) { .\concatStr+ }
Enter 2 strings u want to concatenate
Hello
World!
concatenation: HelloWorld!
```

Overload = operator to carry out string copy

Source Code:

```
#include <iostream>
#include <cstring>

using namespace std;

class equalString{
    public:
        char str[25]; // Classes object of string

        equalString(){} // no parameter constructor

        // Parametrized Constructor
        equalString(char s[]){
            strcpy(this->str, s); // Initialize the string to class object
        }

        bool operator==(equalString s2)
        {
            if (strcmp(str, s2.str) == 0)
                return true;
            else
                return false;
        }
};

int main(){
    char s1[100], s2[100];

    cout << "Enter 2 strings u want to compare and check equal or not" << endl;
    cin >> s1 >> s2;

    equalString a1(s1);
    equalString a2(s2);

    if (a1 == a2) {
        cout << "Strings are equal" << endl;
    }
    else{
        cout << "Strings are not equal" << endl;
```

```
    }
    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
reeha
reeh
Strings are not equal
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
reeha
reeha
Strings are equal
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
apple
app
Strings are not equal
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
apple apple
Strings are equal
PS D:\sem 4\cpp\oops>
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
apple
app
Strings are not equal
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
apple apple
Strings are equal
PS D:\sem 4\cpp\oops>
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strequalornot.cpp -o strequalornot } ; if ($?) { .\strequalornot }
Enter 2 strings u want to compare and check equal or not
a
b
Strings are not equal
PS D:\sem 4\cpp\oops>
```

Overload <= operator to carry out the comparison of strings.

Source Code:

```
#include <cstring>
#include <iostream>
#include <string.h>

using namespace std;

class equalString{
    public:
        char str[25]; // Classes object of string

        equalString(){} // no parameter constructor

        // Parametrized Constructor
        equalString(char s[]){
            strcpy(this->str, s); // Initialize the string to class object
        }

        bool operator<=(equalString s2)
        {
            if (strlen(str) <= strlen(s2.str))
                return true;
            else
                return false;
        }
};

int main(){
    char s1[100], s2[100];

    cout << "Enter 2 strings u want to compare and check equal or not" << endl;
    cin >> s1 >> s2;

    equalString a1(s1);
    equalString a2(s2);

    if (a1 <= a2) {
        cout << "First string is smaller than or equal to second " << endl;
    }
    else{
```

```
        cout << "Second string is smaller than first string" << endl;
    }
    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ comparisonStr.cpp -o comparisonStr } ; if ($?) { .\comparisonStr }
Enter 2 strings u want to compare and check equal or not
reeha
reeh
Second string is smaller than first string
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ comparisonStr.cpp -o comparisonStr } ; if ($?) { .\comparisonStr }
Enter 2 strings u want to compare and check equal or not
firstString
SecndStr
Second string is smaller than first string
PS D:\sem 4\cpp\oops>
```

```
Enter 2 strings u want to compare and check equal or not
abc
ab
Second string is smaller than first string
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ comparisonStr.cpp -o comparisonStr } ; if ($?) { .\comparisonStr }
Enter 2 strings u want to compare and check equal or not
abc
ab
Second string is smaller than first string
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ comparisonStr.cpp -o comparisonStr } ; if ($?) { .\comparisonStr }
Enter 2 strings u want to compare and check equal or not
abc
abcd
First string is smaller than or equal to second
PS D:\sem 4\cpp\oops>
```

Function to display the length of string.

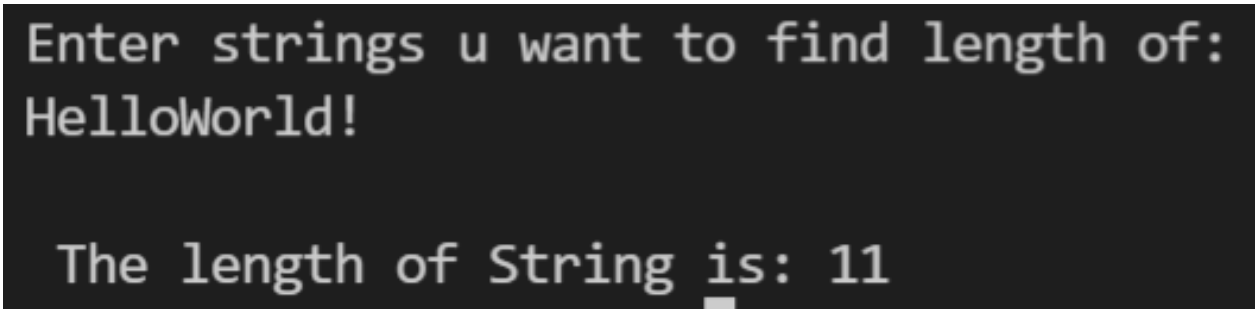
Source code

```
#include <iostream>
#include <cstring>

using namespace std;

class strLength
{
public:
    char str[25]; // Classes object of string

    void strlen()
    {
        cout << "Enter strings u want to find length of:" << endl;
        cin >> str;

        int l = 0;

        for (int i = 0; str[i] != '\0'; i++)
        {
            l++;
        }

        cout << "\n The length of String is: " << l;
    }
};

int main()
{
    strLength obj;
    obj.strlen();
    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strlen.cpp -o strlen } ; if ($?) { .\strlen }
Enter strings u want to find length of:
reeha

 The length of String is: 5
PS D:\sem 4\cpp\oops>
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ strlen.cpp -o strlen } ; if ($?) { .\strlen }
Enter strings u want to find length of:
HelloWorld!

 The length of String is: 11
PS D:\sem 4\cpp\oops>
```



Function tolower() to convert upper case to lower case.

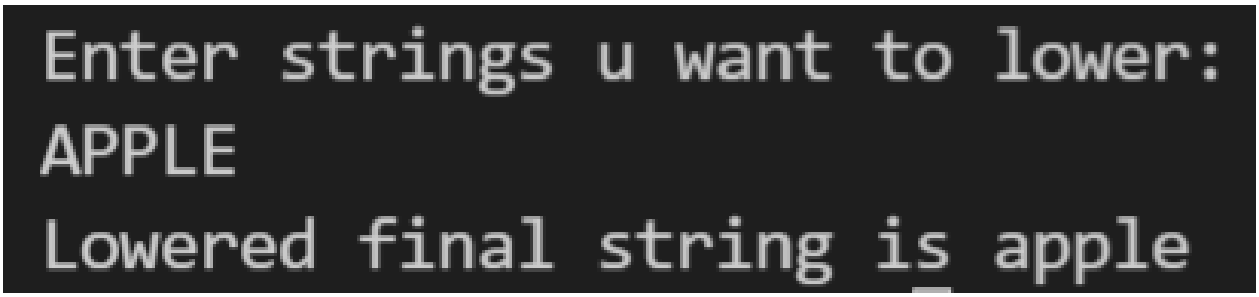
Source code

```
#include <iostream>
#include <cstring>

using namespace std;

class caseChange
{
public:
    char str[25]; // Classes object of string

    void tolower()
    {
        cout << "Enter strings u want to lower:" << endl;
        cin >> str;

        for (int i = 0; str[i] != '\0'; i++)
        {
            if ((str[i] >= 65) && (str[i] <= 90))
            {
                str[i] = str[i] + 32;
            }
        }
        cout << "Lowered final string is " << str;
    }
};

int main()
{
    caseChange obj;
    obj.tolower();
    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to lower:
REEHA
Lowered final string is reeha
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to lower:
APPLE
Lowered final string is apple
PS D:\sem 4\cpp\oops>
```

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to lower:
REEHA
Lowered final string is reeha
```



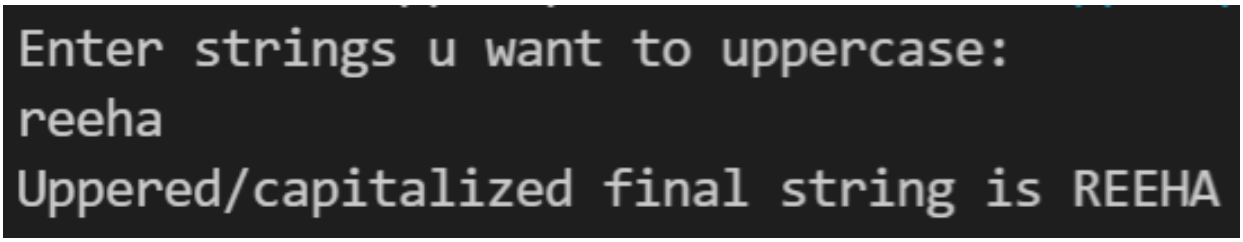
Function toupper() to convert lower case letters to upper case.

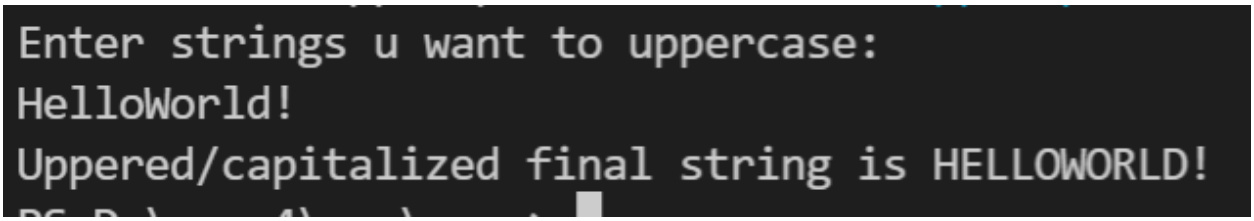
Source code

```
#include <iostream>
#include <cstring>

using namespace std;

class caseChange
{
public:
    char str[25]; // Classes object of string

    void toupper()
    {
        cout << "Enter strings u want to uppercase:" << endl;
        cin >> str;

        for (int i = 0; str[i] != '\0'; i++)
        {
            if ((str[i] >= 97) && (str[i] <= 122))
            {
                str[i] = str[i] - 32;
            }
        }

        cout << "Uppered/capitalized final string is " << str;
    }
};

int main()
{
    caseChange obj;
    obj.toupper();
    return 0;
}
```

Output:

```
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to uppercase:
reeha
Uppered/capitalized final string is REEHA
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to uppercase:
hello
Uppered/capitalized final string is HELLO
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to uppercase:
Hello World!
Uppered/capitalized final string is HELLO
PS D:\sem 4\cpp\oops> cd "d:\sem 4\cpp\oops\" ; if ($?) { g++ tolower.cpp -o tolower } ; if ($?) { .\tolower }
Enter strings u want to uppercase:
HelloWorld!
Uppered/capitalized final string is HELLOWORLD!
PS D:\sem 4\cpp\oops>
```





Source code

```
#include <cstring>
#include <iostream>
#include <string.h>

using namespace std;

class operate
{
    char str[25]; // Classes object of string
public:
    void operator+(char s[25])
    {
        strcat(str, " ");
        strcat(str, s);
    }

    void operator=(char s[25])
    {
        strcpy(str, s);
    }

    void operator<=(char s[25])
    {
        if (strcmp(str, s) > 0)
        {
            cout << str << " is larger than " << s;
        }
        else if (strcmp(str, s) < 0)
        {
            cout << s << " is larger than " << str;
        }
        else
        {
            cout << str << " is equal to " << s;
        }
    }

    void strlen()
    {
```

```
        int l = 0;
        for (int i = 0; str[i] != '\0'; i++)
        {
            l++;
        }
        cout << "\n The length of String is: " << l;
    }
    void toupper()
    {
        for (int i = 0; str[i] != '\0'; i++)
        {
            if ((str[i] >= 97) && (str[i] <= 122))
            {
                str[i] = str[i] - 32;
            }
        }
    }
    void tolower()
    {
        for (int i = 0; str[i] != '\0'; i++)
        {
            if ((str[i] >= 65) && (str[i] <= 90))
            {
                str[i] = str[i] + 32;
            }
        }
    }

    void display()
    {
        cout << "Final string -----------> " << str << endl;
    }
};

int main()
{
    char s[100], st[100], ch;

    cout << "Enter 2 strings u want to compare and check equal or not" << endl;
    cin >> s >> st;

    operate r;
    cout << "\n\nChoose whether u want to \n1)string length \nconcat'+' \n2)to Up
per \ncompare'<' \n3) to lower \ncopy'=' \nexit(x)\n\n";
    //Invoking different operators
```

```
    cin >> ch;
    while (ch != 'x')
    {
        switch (ch)
        {
        case '+':
            r + st;
            r.display();
            break;

        case '=':
            r = st;
            r.display();
            break;

        case '<':
            r <= st;
            break;

        case 1:
            r.strlen();
            break;

        case 2:
            r.toupper();
            r.display();
            break;

        case 3:
            r.tolower();
            r.display();
            break;

        case 'x':
            cout << " Exit";
            exit(0);
        }
        cout << "\n\nChoose whether u want to \n1)string length \nconcat'+' \n2)t
o Upper \ncompare'<' \n3) to lower \ncopy'=' \nexit(x)\n\n";
        //Invoking different operators
        cin >> ch;
    }
    return 0;
}
```

```
// Operator overloading
        void strlen()
        {
            int l=0;
            for(int i=0;str[i]!='\0';i++)
            {
             l++;
            }
            cout<<"\n The length of String is: "<<l;
        }
        void toupper()
        {
            for(int i=0;str[i]!='\0';i++)
            {
             if((str[i]>=97)&&(str[i]<=122))
             {
                str[i]=str[i]-32;
             }
            }
        }
        void tolower()          {
            for(int i=0;str[i]!='\0';i++)
            {
             if((str[i]>=65)&&(str[i]<=90))
             {
                str[i]=str[i]+32;
             }
            }
        }
```

```
void operator+(char s[25])
{
    strcat(str," ");
    strcat(str,s);
}
void operator=(char s[25])
{
    strcpy(str,s);
}
void operator<=(char s[25])
{
    if(strcmp(str,s)>0)
    {
      cout<<str<<" is larger than "<<s;
    }
    else if(strcmp(str,s)<0)
    {
      cout<<s<<" is larger than "<<str;
    }
    else
```

Viva Questions

Q1) What is overloading?

Ans.

C++ allows you to specify more than one definition for a ***function*** *name or an* ***operator*** *in the same scope, which is called* ***function overloading*** *and* ***operator overloading*** *respectively.*

An overloaded declaration is a declaration that is declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and obviously different definition (implementation).

When you call an overloaded ***function*** *or* ***operator****, the compiler determines the most appropriate definition to use, by comparing the argument types you have used to call the function or operator with the parameter types specified in the definitions. The process of selecting the most appropriate overloaded function or operator is called* ***overload resolution****.*

Q2) What are causes of function overloading?

Ans.

When the compiler is unable to decide which function is to be invoked among the overloaded function, this situation is known as **function overloading**.

When the compiler shows the ambiguity error, the compiler does not run the program.

Causes of Function Overloading:

- Type Conversion.
- Function with default arguments.
- Function with pass by reference.